BACKGROUND RENDERING OF IMAGES

Technical Field

[0001] The application relates generally to image processing, and, more particularly, to background rendering of images.

Background

[0002] Image processing can be a computational expensive task that may consume limited hardware resources. Typically, image processing includes the rendering of both a foreground and background of an image. Conventional image processing uses a rendering engine that executes a software application to generate pixel data for the foreground and the background of the images, thereby creating images for display. The foreground of an image is usually a more complex creation in comparison to the background. For example, the background may be as simple as a solid color. However, background rendering can consume limited processing bandwidth of the rendering engine. Such bandwidth could be better used for rendering the more complex foreground parts of the image.

[0003] A typical implementation of a rendering engine uses a back to front rendering, where the background color of a window is processed first using a very fast two dimensional (2D) clear engine. However, a rendering engine based on a front to back rendering implementation generally achieves better anti-aliasing results. With this latter implementation, the background is processed with the slower, normal three dimensional (3D) rendering path used for processing the foreground.

Summary

[0004] Methods, apparatus and systems for background rendering of an image are described. Embodiments of the invention allow for a front to back rendering order for the generating of an image that allows for better anti-aliasing results (relative to back to front rendering). As described in more detail below, embodiments of the invention allow for a front to back rendering without the large time penalties normally associated with the generation of the background color fill. In an embodiment, the background fill information is generated by a hardware logic (such as a field programmable gate array) that is separate from the software being executed within a rendering engine. Accordingly, embodiments of the invention free up the bandwidth of the rendering engine, thereby allowing for the rendering of more complex foreground image (without the time penalties associated therewith). Moreover, this separate hardware logic merges the background fill data with the foreground data to form the final image. In an embodiment, this separate hardware logic allows for the merging of a background video and/or a background color with the foreground image rendered by the rendering engine.

[0005] In one embodiment, an apparatus includes a rendering engine to render a foreground of an image. The apparatus also includes a logic, separate from the rendering engine, to merge at least one background color with the foreground of the image.

Brief Description of the Drawings

[0006] Embodiments of the invention may be best understood by referring to the following description and accompanying drawings which illustrate such embodiments. The numbering scheme for the Figures included herein are such that the leading number for a given reference number in a Figure is associated with the number of the Figure. For example, an apparatus 100 can be located in Figure 1. However, reference numbers are the same for those elements that are the same across different Figures. In the drawings:

[0007] **Figure 1** illustrates an apparatus that provides for the merging of background colors with a rendered foreground image, according to one embodiment of the invention.

[0008] **Figure 2** illustrates a background color table used for merging a background color with a rendered foreground image, according to one embodiment of the invention.

[0009] **Figure 3** illustrates a flow diagram for rendering an image, according to one embodiment of the invention.

[0010] **Figure 4** illustrates a flow diagram for blending the background fill color with the rendered foreground image, according to one embodiment of the invention.

[0011] **Figure 5** illustrates a system that merges a background color with a rendered foreground image, according to one embodiment of the invention.

Detailed Description

[0012] Methods, apparatuses and systems for background rendering of images are described. In the following description, numerous specific details such as logic implementations, opcodes, means to specify operands, resource partitioning/sharing/duplication implementations, types and interrelationships of system components, and logic partitioning/integration choices are set forth in order to provide a more thorough understanding of the present invention. It will be appreciated, however, by one skilled in the art that embodiments of the invention may be practiced without such specific details. In other instances, control structures, gate level circuits and full software instruction sequences have not been shown in detail in order not to obscure the embodiments of the invention. Those of ordinary skill in the art, with the included descriptions will be able to implement appropriate functionality without undue experimentation.

[0013] References in the specification to "one embodiment", "an embodiment", "an example embodiment", etc., indicate that the embodiment described may include a particular feature, structure, or characteristic, but every embodiment may not necessarily include the particular feature, structure, or characteristic. Moreover, such phrases are not necessarily referring to the same embodiment. Further, when a particular feature, structure, or characteristic is described in connection with an embodiment, it is submitted that it is within the knowledge of one skilled in the art to affect such feature, structure, or characteristic in connection with other embodiments whether or not explicitly described.

[0014] Embodiments of the invention include features, methods or processes embodied within machine-executable instructions provided by a machine-readable medium. A machine-readable medium includes any mechanism which provides (i.e., stores and/or transmits) information in a form accessible by a machine (e.g., a computer, a network device, a personal digital assistant, manufacturing tool, any device with a set of one or more processors, etc.). In an exemplary embodiment, a machine-readable medium includes volatile and/or non-volatile media (e.g., read only memory (ROM), random access memory (RAM), magnetic disk storage media, optical storage media, flash memory devices, etc.), as well as electrical, optical, acoustical or other form of propagated signals (e.g., carrier waves, infrared signals, digital signals, etc.)).

[0015] Such instructions are utilized to cause a general or special purpose processor, programmed with the instructions, to perform methods or processes of the embodiments of the invention. Alternatively, the features or operations of embodiments of the invention are performed by specific hardware components which contain hard-wired logic for performing the operations, or by any combination of programmed data processing components and specific hardware components. Embodiments of the invention include software, data processing hardware, data processing system-implemented methods, and various processing operations, further described herein.

[0016] A number of figures show block diagrams of systems and apparatus for background rendering of images, in accordance with embodiments of the invention. A number of figures show flow diagrams illustrating operations for background rendering of images. The operations of the flow diagrams will be described with references to the systems/apparatus shown in the block diagrams. However, it should be understood that the operations of the flow diagrams could be performed by embodiments of systems and apparatus other than those discussed with reference to the block diagrams, and embodiments discussed with reference to the systems/apparatus could perform operations different than those discussed with reference to the flow diagrams.

[0017] **Figure 1** illustrates an apparatus that provides for the merging of background colors with a rendered foreground image, according to one embodiment of the invention. Figure 1 illustrates an apparatus 100 that includes a rendering engine 102, a frame buffer 104, a background merge logic 106, a video source 108 and a display monitor 110. The frame buffer 104 includes an A buffer 150, a B buffer 152 and a Z buffer 154. The background merge logic 106 includes a saturation enable logic 112, a background color table 114, a control logic 115, a video lookup table 116, a graphic lookup table 118, a multiply logic 120, an add logic 122, gamma/clamping tables 124, a video FIFO 160 and a buffer select table 156. Although not shown for sake of clarity, the control logic 115 is coupled to the saturation enable logic 112, the background color table 114, the video lookup table 116, the graphic lookup table 118, the multiply logic 120, the add logic 122, the gamma/clamping tables 124, the video FIFO 160 and the buffer select table 156.

[0018] The rendering engine 102 generates a rendered image (that includes the color values for the foreground pixels that comprise the rendered image). The rendering engine 102 stores colors values (red, green, blue, alpha, window identification) for the foreground pixels in the A buffer 150, the B buffer 152 and the Z buffer 154. The A buffer 150 and the B buffer 152 are ping-pong type buffers. In particular, the rendering

engine 102 is writing to one of these buffers (current write buffer), while the other buffer is being read from for display (current read buffer). For each pixel, the A buffer 150 and the B buffer 152 include alpha, red, green and blue (ARGB) intensity values. The alpha intensity value specifies the amount of an independent pixel source to be merged with the image rendered by the rendering image 102 included in the red, green and blue pixel values.

[0019] In one embodiment, an image may be made of a number of independently rendered smaller regions (windows). Each window represents a part of the overall displayed image. The Z buffer 154 includes a number of entries, wherein a given entry includes an identification of a window for a given pixel in an image. For example, in an embodiment, an image may include 16 different windows. As further described below, the lookup into the Z buffer 154 provides an identification of a window within which a given pixel is located. A lookup into the buffer select table 156 based on this window identification is performed to select either the A buffer 150 or the B buffer 152 for display (also referred to as front buffer detection). Therefore, a given pixel in a display is retrieved from either the A buffer 150 or the B buffer 152. Accordingly, in an embodiment wherein an image is partitioned into 16 windows, the buffer select table 156 is a 16-location, 1-bit wide look-up table. In one such embodiment, the buffer select table 156 includes 16 different entries that include a single bit to identify the A buffer 150 or the B buffer 152. In one embodiment, the values stored in the buffer select table 156 are updated at the completion of a given scene being processed by the background merge logic 106.

[0020] The saturation enable logic 112 is coupled to receive alpha intensity values 130 from the current read buffer for the pixels to be displayed. The saturation enable logic 112 is coupled to output background attenuation 132, which is inputted into the multiply logic 120. In one embodiment, the saturation enable logic 112 either inverts the alpha intensity values 130 (1 – alpha) or passes the alpha intensity values 130 for

special blending modes. In an embodiment, the attenuation value 132 represents a value in a range of zero to one.

[0021] The video source 108 is coupled to input video 134 into a video FIFO 160. In an embodiment, the video FIFO 160 is partitioned into two banks. After writing a frame of the video 134 into a first bank of the video FIFO 160, the control logic 115 causes the next frame of the video 134 to be written to the second bank, then writing to the first bank, etc.

[0022] The video FIFO 160 is coupled to the video lookup table 116. In one embodiment, the background color table 114 includes a number of entries for storage of background color data for each active window. In an embodiment, the background color table 114 includes 16 entries. A given entry therein is associated with a window in the image. An entry includes an identification of a window and the color values for the window. One embodiment of the background color table 114 is illustrated in Figure 2, which is described in more detail below.

[0023] In one embodiment, the video lookup table 116 or the background color table 114 are coupled to input a background color 136 into the multiply logic 120. In an embodiment, the video lookup table 116 and the background color table 114 are coupled to input a background color 136 into the multiply logic 120. In one such embodiment, an application executing external to the background merge logic 106 configures video position registers internal to the control logic 115 (not shown). For example, in the system 500 of Figure 5, the processor therein may execute a graphics application that causes the generation of instructions that are stored in the background merge logic 106 to allow for various control and configuration of the background merge logic 106 (including the setting of these video position registers). The configuration of these video position registers define where video is selected for display. Accordingly, in an embodiment, the control logic 115 enables the display of background color (from the background color

table 114) at locations on the display where video (from the video source 108) is not selected for display.

[0024] The control logic 115 causes a background color 136 to be input into the multiply logic 120 from the video lookup table 116 and/or the background color table 114. The control logic 115 includes a number of control registers for controlling the merging of the video 134 or the color values of the background pixels from the background color table 114 with the foreground image separately rendered by the rendering engine 102. The multiply logic 120 outputs an adjusted background color 137 based on the background color 136 and the background attenuation 132. The adjusted background color 137 is inputted into the add logic 122.

[0025] The color values of the rendered image 138 are inputted from the current read buffer into the graphic lookup table 118 and the add logic 122. As further described below, in one embodiment, the control logic 115 performs smooth shading for these color values based on a lookup into the graphic lookup table 118. As shown, a lookup is performed into the graphic lookup table 118 to output a value that has been smooth shaded based on the color values retrieved from the current read buffer. Moreover, as shown, this lookup may be bypassed, thereby allowing for direct input of the color values for the foreground pixel into the add logic 122. The add logic 122 is coupled to output the final image 140 to the gamma/clamping tables 124. The gamma/clamping tables 124 are coupled to output a resulting image 140 to the display monitor 110.

[0026] As further described below, in one embodiment, the background merge logic 106 merges a background color with the rendered foreground color based on a pre-multiply of the background color with the alpha intensity value for the foreground pixel to allow for an alpha source saturate blending. The adjusted background color is added to the color data (red, green and blue, respectively) for the selected foreground pixel as illustrated by equations (1)-(3) (which includes an alpha source saturation blending):

[0027] (1) $R_{RESULT} = R_{BACKGROUND} * (1\text{-}ALPHA_{SRC}) + R_{SRC}$

[0028] (2) $G_{RESULT} = G_{BACKGROUND} * (1\text{-}ALPHA_{SRC}) + G_{SRC}$

[0029] (3) $B_{RESULT} = B_{BACKGROUND} * (1\text{-}ALPHA_{SRC}) + B_{SRC}$

[0030] While Figure 1 illustrates the merging of a video or a background color with the rendered foreground image, embodiments of the invention are not so limited. In another embodiment, video is not merged with the rendered foreground image. Therefore, the apparatus 100 does not include the video source 108 coupled to input the video 134 into the video FIFO 160 and does not include the video lookup table 116. Accordingly, the background merge logic 106 merges different types of background colors (and not video). The operations of the apparatus 100 are described in more detail below in conjunction with the flow diagrams 300 and 400 of Figures 3 and 4, respectively.

[0031] **Figure 2** illustrates a background color table used for merging a background color with a rendered foreground image, according to one embodiment of the invention. In particular, Figure 2 illustrates one embodiment of the background color table 114. As shown, the background color table 114 maintains background color values for the different possible windows that may be displayed. Accordingly, the background colors in the background color table 114 are managed in a double buffered scheme similar to that of the A buffer 150 and the B buffer 152. Therefore, for a given entry, there are two color values: background color A and background color B associated with each window in the image to be displayed. The background color A merges with the corresponding pixel from the A buffer 150. The background color B merges with the corresponding pixel from the B buffer 152.

[0032] In one embodiment, one of the window identifications is reserved for pixels not actively populated. Therefore, if merge is to occur for a region of an image that is not associated with a window identification therein, this reserve window

identification is used. In one embodiment, the color value for this reserved window identification is black. In an embodiment, logic (applications/hardware, etc.) external to the background merge logic 106 cannot modify the entry for this reserved window identification in the background color table 114. In one embodiment, logic (applications/hardware, etc.) external to the background merge logic 106 updates the colors stored in the background color table 114.

[0033] In one embodiment, for each window of a frame of the image, logic (applications/hardware, etc.) external to the background merge logic 106 stores the background color in the corresponding A-background entry for this window in the background color table 114, while the background color in the corresponding B-background entry for this window is being displayed and vice versa. Accordingly, this allows the updating of the next frame's background color for the different windows without interfering with the display of the current frame.

[0034] Moreover, as described in more detail below, the window identification stored in the Z buffer 154 and the A/B buffer selection stored in the buffer select table 156 are used to address the background color table 114 in a table lookup approach. In other words, the Z buffer 154 and the buffer select table 156 are used to select the location of the color values of the background color to be merged with the different parts of the rendered foreground image for display. One embodiment of logic (applications/hardware, etc.) to load the color values into the background color table 114 is described in more detail below in conjunction with the system 500 of Figure 5.

[0035] One embodiment of the operations of the apparatus 100 is now described. In particular, **Figure 3** illustrates a flow diagram for rendering an image, according to one embodiment of the invention.

[0036] In block 302 of the flow diagram 300, the foreground of an image is rendered by a rendering engine. With reference to the embodiment of Figure 1, the rendering engine 102 renders the foreground of an image. In an embodiment, the

foreground may be different types of symbology. For example, the foreground may be text, a cursor, etc. In an embodiment, the rendering engine 102 generates the color values (e.g., red, green, blue, and alpha) for the different pixels in the foreground. Additionally, in an embodiment, the rendering engine 102 generates an identification of a window on the display screen wherein the pixel is to be displayed. In particular, the display screen is partitioned into a number of windows for processing of the pixels. Accordingly, the rendering engine 102 generates the identification of one of such windows within which the pixel is located.

[0037] One embodiment of a system that includes the apparatus 100 is described in more detail below in conjunction with Figure 5. As described, a processor in the system generates and stores a number of instructions into a system memory. Such instructions control the rendering of the image by the rendering engine 102. Therefore, the rendering engine 102 retrieves these instructions from the system memory. Based on such instructions, the rendering engine 102 renders the foreground of the image. In one embodiment, software executing on a processor internal to the rendering engine 102 executes these instructions for rendering the foreground of the image. Control continues at block 304.

[0038] In block 304, the color values and the window identifications of the foreground pixels are stored in a frame buffer, by the rendering engine. With reference to the embodiment of Figure 1, the rendering engine 102 stores the color values and the window identifications for the different foreground pixels in the frame buffer 104. As described above, the A buffer 150 and the B buffer 152 (in the frame buffer 104) are ping-pong type buffers. The rendering engine 102 stores the color values of the foreground to the current write buffer (either the A buffer 150 or the B buffer 152). In one embodiment, each foreground pixel has an associated entry in the current write buffer. Such an entry includes an alpha intensity value, a red value, a green value and a

blue value. The rendering engine 102 also stores the window identification of the foreground pixel into the Z buffer 154. Control continues at block 306.

[0039] In block 306, the color values and the window identifications of the foreground pixels are retrieved, by logic that is separate from the rendering engine. With reference to the embodiment of Figure 1, the background merge logic 106 retrieves the color value and the window identifications of the foreground pixels from the current read buffer (either the A buffer 150 or the B buffer 152) and from the Z buffer 154, respectively. As shown, the control logic 115 retrieves the color values of the rendered image 138, which are input into the graphic lookup table 118 and the add logic 122. The control logic 115 also retrieves the alpha intensity values 130 of the color values, which are input into the saturation enable logic 112. Control continues at block 308.

[0040] In block 308, a determination is made of whether the background is video. With reference to the embodiment of Figure 1, the control logic 115 determines whether the background is video based on a Boolean value (stored in one of the control registers internal to the control logic 115) that includes whether to merge video with the rendered foreground image. This Boolean value may be configured by an application executing external to the background merge logic 106. For example, in the system 500 of Figure 5, the processor therein may execute a graphics application that causes the generation of instructions that are stored in the system memory. Such instructions may be retrieved by the background merge logic 106 to allow for various control and configuration of the background merge logic 106 (including the setting of this Boolean value).

[0041] In block 310, upon determining that the background is video, the video is blended with the rendered foreground image. With reference to the embodiment of Figure 1, the background merge logic 106 blends the video with the rendered foreground image. Control continues at block 314, which is described in more detail below.

[0042] In block 312, upon determining that the background is not video, the background fill data is blended with the rendered foreground image. With reference to

the embodiment of Figure 1, the background merge logic 106 blends the background color values with the rendered foreground image. A more detailed description of the operations for blending the background color values with the rendered foreground image is described in more detail below in conjunction with the flow diagram 400 of Figure 4. Control continues at block 314.

[0043] In block 314, the resulting image is output for display. With reference to the embodiment of Figure 1, the background merge logic 106 outputs the resulting image 140 for display to the display monitor 110. The operations of the flow diagram 300 may continue at block 302, as a different rendered foreground image is merged with the background video and/or the background colors.

[0044] The operations for blending the background fill color with the rendered foreground image are now described. In particular, **Figure 4** illustrates a flow diagram for blending the background fill color with the rendered foreground image, according to one embodiment of the invention. For sake of clarity, the flow diagram 400 of Figure 4 illustrates this blending operation for a single pixel in the image to be output for display. Accordingly, the operations of the flow diagram 400 may be performed for the different pixels in a given image to be output for display.

[0045] In block 402, based on an identification of the window that includes the foreground pixel to be processed, an identification of a current read buffer is retrieved. With reference to the embodiment of Figure 1, the control logic 115 retrieves an identification of the current read buffer (either the A buffer 150 or the B buffer 152) based on the identification of the window for a pixel. In particular, the control logic 115 retrieves the identification of the window for a pixel from the Z buffer 154. Moreover, the control logic 115 performs a lookup into the buffer select table 156 that identifies either the A buffer 150 or the B buffer 152 as the current read buffer for the pixel. Control continues at block 404.

[0046] In block 404, color values of the background pixel located at the same location in the image as the foreground pixel (being processed) are retrieved. With reference to the embodiment of Figure 1, the control logic 115 retrieves the color values of the background pixel from the background color table 114 or the video lookup table 116 based on the window identification and the current buffer selection (the A buffer 150 or the B buffer 152). The control logic 115 causes the background color 136 to be input into the multiply logic 120 from the background color table 114 or the video lookup table 116. Returning to Figure 2 to illustrate the retrieval from the background color table 114, the control logic 115 selects the entry therein based on the identification of the window and the identification of the current read buffer (either the A buffer 150 or the B buffer 152). Control continues at block 406.

[0047] In block 406, an intensity of the color values of the background pixel is adjusted based on the alpha intensity value of the foreground pixel. With reference to the embodiment of Figure 1, the control logic 115 retrieves the alpha intensity value (associated with the foreground pixel from the current read buffer), which is input into the saturation enable logic 112. The saturation enable logic 112 outputs the background attenuation 132 based on the alpha intensity value. In an embodiment, there are two blending modes: alpha source saturate and alpha blending (source alpha or 1 – source alpha). Accordingly, if the background merge logic 106 is configured to be in the alpha source saturate blending mode, the saturation enable logic 112 outputs the background attenuation 132 that provides alpha source saturate blending. If the background merge logic 106 is configured to not be in the alpha source saturate blending mode, the saturation enable logic 112 outputs the background attenuation 132 with alpha blending. Accordingly, the two different modes (alpha source saturate and alpha blending) allow for the use of "alpha" or "1-alpha", respectively, for the blending operations.

[0048] The background attenuation 132 is inputted into the multiply logic 120. The multiply logic 120 adjusts the background color 136 based on the value of the

background attenuation 132. As illustrated by equations (1) – (3) (set forth above for alpha source saturate blending), the background attenuation 132 has a value of '1 - $ALPHA_{SRC}$'. The multiply logic 120 multiplies the background attenuation 132 by the background color 136 (for each of the red, green and blue background colors). Control continues at block 408.

[0049] In block 408, the adjusted color values of the background pixel are blended with the color values of the foreground pixel. With reference to the embodiment of Figure 1, the control logic 115 retrieves the color values of the foreground pixel from the current read buffer (shown as the color values of the rendered image 138). In one embodiment, the control logic 115 smoothes shading for these color values based on a lookup into the graphic lookup table 118. As shown, a lookup is performed into the graphic lookup table 118 to output a value that has been smooth shaded based on the color values retrieved from the current read buffer. Moreover, as shown, this lookup may be bypassed, thereby allowing for direct input of the color values for the foreground pixel into the add logic 122. The adjusted background color 137 (which is the result of the background color adjustment operation) is input into the add logic 122.

[0050] The add logic 122 blends the color values of the foreground pixel with the color values of the adjusted background color 137. As illustrated by equations (1) – (3) (set forth above), the add logic 122 adds the red, green and blue value of the foreground pixel to the red, green and blue value of the adjusted background color 137, respectively. Moreover, in one embodiment, the control logic 115 clamps the values of the result of this blend operation to a predetermined number of bits based on the clamping tables in the gamma/clamping tables 124.

[0051] In an embodiment, the control logic 115 performs gamma correction of the values of the result of this blend operation based on the gamma table in the gamma/clamping tables 124. In one embodiment, the video 134 that is input into the background merge logic 106 has a gamma value of approximately 0.45. The video 134 is

converted into a linear space during the operations within the background merge logic 106. Therefore, the resulting image 140 is converted from a linear state back to an image having a gamma of approximately 0.45 based on the gamma table in the gamma/clamping tables 124.

[0052] While embodiments of the invention may operate in a number of different systems, one embodiment is now described. In particular, **Figure 5** illustrates a system that merges a background color with a rendered foreground image, according to one embodiment of the invention. In particular, Figure 5 illustrates a system 500 that includes a processor 502, a system memory 504, a bridge logic 506, the rendering engine 102, the frame buffer 104, the background merge logic 106, the video source 108 and the display monitor 110. The processor 502 and the system memory 504 are coupled to the bridge logic 506. The rendering engine 102 is coupled to the bridge logic 506 through a graphics bus 520. The rendering engine 102 is coupled to the frame buffer 104. The background merge logic 106 is coupled to the bridge logic 522 and the frame buffer 104 through a system bus 522. The background merge logic 106 is coupled to the display monitor 110. The background merge logic 106 is coupled to the video source 108 through a video bus 524.

[0053] In one embodiment, the processor 502 executes instructions of a graphics application that generates graphics instructions that are stored into the system memory 504 through the bridge logic 506. The rendering engine 102 retrieves at least a part of these graphics instructions and renders foreground images (to be displayed on the display monitor 110) based on such instructions. The rendering engine 102 stores color values of these rendered foreground images into the frame buffer 104. The background merge logic 106 retrieves at least a part of these graphic instructions (stored in the system memory 504) for its configuration. For example, such graphics instructions may update the colors in the background color table 114. Additionally, the background merge logic 106 retrieves at least a part of these graphics instructions and merges video (from the

video source 108) and/or background fill colors in the background color table 114 with the rendered foreground images (retrieved from the frame buffer 104) based on these instructions. In an embodiment, such instructions direct logic in the background merge logic 106 to use the other buffer (e.g., the A buffer 150 if the B buffer 152 is the current read buffer or vice versa) when the processing of data for a given scene being displayed has completed.

[0054] Thus, methods, apparatuses and systems for background rendering of images have been described. Although the present invention has been described with reference to specific exemplary embodiments, it will be evident that various modifications and changes may be made to these embodiments without departing from the broader spirit and scope of the invention. Therefore, the specification and drawings are to be regarded in an illustrative rather than a restrictive sense.